Никита Культин

Delphi
в задачах и примерах

2-е издание

Санкт-Петербург
«БХВ-Петербург»
2008

УДК 681.3.068+800.92Delphi
ББК 32.973.26-018.1
К90

Культин Н. Б.

К90 Delphi в задачах и примерах. — 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2008. — 288 с.: ил. + CD-ROM

ISBN 978-5-94157-997-6

Книга представляет собой сборник примеров программ и задач для самостоятельного решения в среде Delphi. Примеры и задачи различной сложности — от простейших до приложений работы с графикой, мультимедиа и базами данных — демонстрируют назначение компонентов, раскрывают тонкости разработки в Delphi. Справочник, входящий в книгу, содержит описание базовых компонентов и часто используемых функций. Во втором издании обновлены старые и добавлены новые примеры. На прилагаемом компакт-диске находятся проекты, представленные в книге.

Для начинающих программистов

УДК 681.3.068+800.92Delphi
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Екатерина Капалыгина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Игоря Цырульникова*
Оформление обложки	*Елены Беляевой*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.07.08.
Формат 60×90 1/16. Печать офсетная. Усл. печ. л. 18.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.60.953.Д.003650.04.08 от 14.04.2008 г. выдано Федеральной службой по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-94157-997-6

Оглавление

Предисловие

В последнее время резко возрос интерес к программированию. Это связано с развитием и внедрением в повседневную жизнь информационно-коммуникационных технологий. Если человек имеет дело с компьютером, то рано или поздно у него возникает желание, а иногда и необходимость программировать.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую "быструю разработку". В основе идеологии систем быстрой разработки (RAD-систем, Rapid Application Development — среда быстрой разработки приложений) лежат технологии визуального проектирования и событийного объектно-ориентированного программирования, суть которых заключается в том, что среда разработки берет на себя большую часть рутины, оставляя программисту работу по конструированию диалоговых окон и созданию функций обработки событий. Производительность программиста при использовании RAD-систем — фантастическая!

Среди RAD-систем особо выделяется среда Borland Delphi, которая позволяет создавать различные программы: от простейших однооконных приложений до программ управления распределенными базами данных. В качестве языка программирования в среде Borland Delphi используется язык Delphi (Delphi language), являющийся прямым потомком хорошо известного всем программистам языка Pascal.

Чтобы научиться программировать, надо программировать — писать программы, решать конкретные задачи. Для этого необходимо изучить язык программирования и среду разработки. Осво-

ить язык программирования Delphi не очень сложно. Труднее изучить среду программирования, точнее научиться использовать компоненты. И здесь хорошим подспорьем могут быть программы, которые демонстрируют назначение компонентов и особенности их применения.

В книге, которую вы держите в руках, собраны разнообразные примеры, которые не только демонстрируют возможности среды разработки Delphi, но и знакомят с принципами работы с графикой, звуком, базами данных. Следует обратить внимание, что большинство примеров не являются учебными в чистом смысле, это вполне работоспособные программы.

Книга состоит из двух частей и приложения.

Первая часть содержит примеры и задачи для самостоятельного решения. Примеры представлены в виде краткого описания, сформулированного в форме задания для самостоятельного решения, диалоговых окон и хорошо документированных текстов программ. Для простых задач рассмотрены только функции обработки событий. Текст остальных программ приведен полностью.

Вторая часть книги — это краткий справочник по языку программирования Delphi. В нем можно найти описание свойств компонентов, использованных в приведенных примерах.

Научиться программировать можно, только программируя, решая конкретные задачи. При этом достигнутые в программировании успехи в значительной степени зависят от опыта. Поэтому, чтобы получить максимальную пользу от книги, вы должны работать с ней активно. Изучайте листинги, старайтесь понять, как работают программы. Не бойтесь экспериментировать — вносите изменения в программы.

Если что-то не понятно, обратитесь к справочнику *(часть 2)*, справочной системе Delphi или литературе, например: **Культин Н. Б. Основы программирования в Delphi 7. — СПб.: БХВ-Петербург, 2008**. В ней, помимо описания языка программирования и среды разработки Delphi, компонентов, процессов создания и отладки программ, вы найдете ответы на многие вопросы, в том числе: как при помощи Microsoft Help Workshop сформировать файл справки или, используя Install-Shield Express, создать установочный CD-ROM.

Часть 1

Примеры и задачи

БАЗОВЫЕ КОМПОНЕНТЫ

В этом разделе приведены простые примеры и задачи, основное назначение которых — научить работать с базовыми компонентами.

Общие замечания

- Процесс создания программы в Delphi состоит из двух шагов: сначала нужно создать форму (диалоговое окно), затем — написать процедуры обработки *событий*. Форма *приложения* (так принято называть прикладные программы, работающие в Windows) создается путем добавления в форму *компонентов* и последующей их настройки.
- В форме практически любого приложения есть компоненты, которые обеспечивают интерфейс (взаимодействие) между программой и пользователем. Такие компоненты называют *базовыми*. К базовым компонентам можно отнести:
  - `Label` — поле вывода текста;
  - `Edit` — поле ввода/редактирования текста;
  - `Button` — командную кнопку;
  - `CheckBox` — независимую кнопку выбора;
  - `RadioButton` — зависимую кнопку выбора;
  - `ListBox` — список выбора;
  - `ComboBox` — комбинированный список выбора.

- Вид компонента, его размер и поведение определяются значениями *свойств* (характеристик) компонента (описание свойств базовых компонентов можно найти в справочнике, во второй части книги).
- Основную работу в программе выполняют процедуры обработки *событий* (описание основных событий можно найти в справочнике, во второй части книги).
- Исходную информацию программа может получить из полей ввода/редактирования (компонент `Edit`), списка выбора (компонент `ListBox`) или комбинированного списка (компонент `ComboBox`). Для ввода значений логического типа можно использовать компоненты `CheckBox` и `RadioButton`.
- Результат программа может вывести в поле вывода текста (компонент `Label`) или в окно сообщения (функция `MessageDlg`).
- Для преобразования текста, например находящегося в поле ввода/редактирования, в целое число нужно использовать функцию `StrToInt`, а в дробное — функцию `StrToFloat`. Для преобразования целого, например значения переменной, в строку нужно использовать функцию `IntToStr`, а для преобразования дробного — функцию `FloatToStr` или `FloatToStrF`.

1. Написать программу **Мили-километры**, которая пересчитывает расстояние из миль в километры. Рекомендуемый вид формы приведен на рис. 1.1.

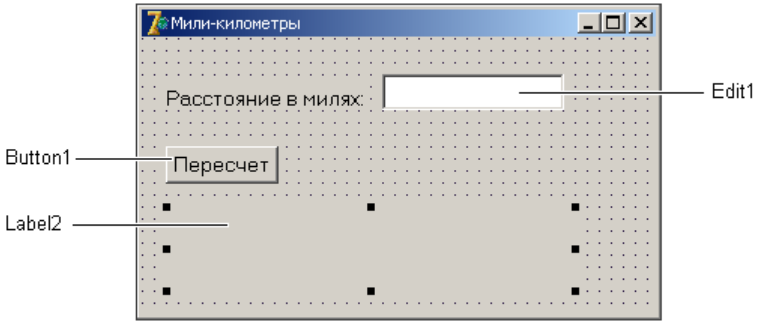

Рис. 1.1. Форма программы **Мили-километры**

```
// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
  mile: real;  // расстояние в милях
  km: real;    // расстояние в километрах

begin
  // ввести исходные данные
  mile := StrToFloat(Edit1.Text);

  // пересчитать
  km := mile * 1.609344; // 1 миля - 1,609344 км

  // вывести результат
  Label2.Caption := FloatToStr(mile) + ' миль - это ' +
                    FloatToStr(km) + ' км.';
end;
```

2. Усовершенствуйте программу **Мили-километры** так, чтобы пользователь мог ввести в поле **Расстояние** только число.

```
// нажатие клавиши в поле компонента Edit1
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
  // Процедура проверяет, является ли символ, соответствующий
  // нажатой клавише (Key), допустимым. Если символ неверный,
  // то он заменяется "нуль символом",
  // который в поле редактирования не отображается.
  // В результате у пользователя создается впечатление,
  // что клавиатура не реагирует на нажатие "неверных"
  // клавиш. В данном случае правильными являются цифровые
  // клавиши, запятая и <Backspace>.

  case Key of
    '0' ..'9', #8: ;  // цифра или <Backspace>
              ',':    // запятая
          if Pos(',',Edit1.Text) <> 0 // запятая уже введена
                      then Key := #0;
    else  Key := #0; // остальные символы не отображать
  end;
end;
```

```
// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
  mile: real;  // расстояние в милях
  km: real;    // расстояние в километрах
begin
  // Если в поле Edit1 нет данных, то при выполнении
  // функции StrToFloat возникает исключение (ошибка).
  // Проверим, введены ли исходные данные.

  if Length(Edit1.Text) = 0 then
  begin
    ShowMessage('Надо ввести исходные данные');
    exit;
  end;

  // пользователь ввел расстояние в милях
  mile := StrToFloat(Edit1.Text);

  km := mile * 1.609344; // 1 миля - 1,609344 км

  Label2.Caption := FloatToStr(mile) + ' миль - это ' +
                    FloatToStrF(km,ffFixed,6,2) + ' км.';
end;
```

3. Написать программу **Конвертор**, которая пересчитывает цену из долларов в рубли. Рекомендуемый вид формы приведен на рис. 1.2. Программа должна быть спроектирована таким образом, чтобы пользователь мог ввести в поля редактирования только дробные числа. При нажатии клавиши <Enter> в поле **Курс** курсор должен переходить в поле **Цена**, а при нажатии этой же клавиши в поле **Цена** — на кнопку **Пересчет**.

```
// нажатие  клавиши в поле Курс
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
    case Key of
        '0'..'9', #8: ;  // цифры и <Backspace>
        '.',',':
             // Обработку десятичного разделителя
             // сделаем "интеллектуальной". Заменим точку и
```

```
            // запятую на символ DecimalSeparator - символ,
            // который при текущей настройке операционной
            // системы должен использоваться
            // при записи дробных чисел.
            begin
               Key := DecimalSeparator;
               // проверим, введен ли уже в поле
               // Edit десятичный разделитель
               if pos(DecimalSeparator,Edit1.Text) <> 0
                  then Key := #0;
            end;
         #13: Edit2.SetFocus; // Нажата клавиша <Enter> -
                              // переместить курсор
                              // в поле Edit2
                              // остальные символы запрещены
         else Key := #0;
     end;
end;

// нажатие клавиши в поле Цена
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
    case Key of
        '0'..'9', #8: ;  // цифры и <Backspace>
        '.',',':
            begin
               Key := DecimalSeparator;
               // проверим, введен ли уже в поле
               // Edit десятичный разделитель
               if pos(DecimalSeparator,Edit1.Text) <> 0
                  then Key := #0;
            end;
        // Сделать активной кнопку Пересчет
        #13: Button1.SetFocus;
        else Key := Char(0);   // остальные символы запрещены
     end;
end;

// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
```

```
var
  usd: real;    // цена в долларах
  k:   real;    // курс
  rub: real;    // цена в рублях

begin
    k := StrToFloat(Edit1.Text);
    usd := StrToFloat(Edit2.Text);

    // пересчитать цену из долларов в рубли
    rub := usd * k;

    // вывести результат расчета в поле Label4
    Label4.Caption := FloatToStr(usd) + '$ = ' +
                      FloatToStrF(rub, ffCurrency, 6,2);
end;
```

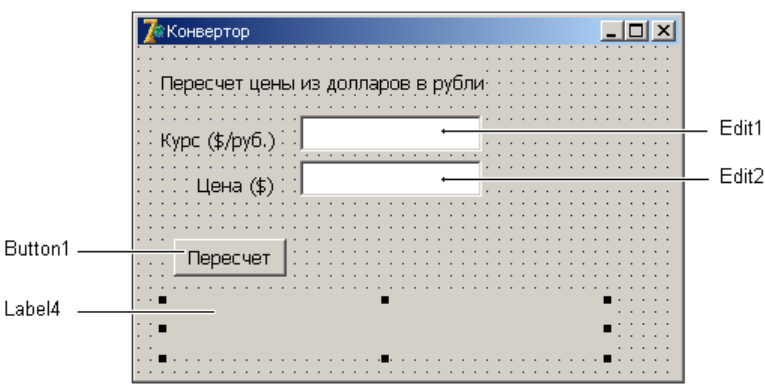

Рис. 1.2. Форма программы **Конвертор**

4. Усовершенствуйте программу **Конвертор** так, чтобы событие `KeyPress` обоих полей редактирования обрабатывала одна процедура, а также чтобы кнопка **Пересчет** становилась доступной только после ввода данных в оба поля редактирования.

```
{
  Процедура EditKeyPress обрабатывает нажатие клавиш
  в полях Курс и Цена. Сначала надо обычным образом
  создать процедуру обработки события KeyPress для поля Edit1,
```

```
  назвав ее EditKeyPress. Затем надо выбрать компонент Edit2
  и указать процедуру EditKeyPress в качестве процедуры
  обработки события KeyPress.
  Чтобы узнать, на каком компоненте произошло
  событие, надо проверить значение свойства Sender.
}
procedure TForm1.EditKeyPress(Sender: TObject; var Key: Char);
begin
    case Key of
        '0'..'9', #8: ;  // цифры и <Backspace>

        '.',',':
             // Обработку десятичного разделителя
             // сделаем "интеллектуальной". Заменим точку и
             // запятую на символ DecimalSeparator - символ,
             // который при текущей настройке операционной
             // системы должен использоваться
             // при записи дробных чисел.
             begin
                Key := DecimalSeparator;
                // проверим, введен ли уже в поле
                // Edit десятичный разделитель
                if pos(DecimalSeparator,Edit1.Text) <> 0
                   then Key := #0;
             end;
         #13: // клавиша <Enter>
              // параметр Sender содержит имя компонента,
              // на котором произошло событие
              if Sender = Edit1 then
                 // Переместить курсор в поле Edit2
                 Edit2.SetFocus
             // Установить фокус на Button1
             else Button1.SetFocus;
         else Key := #0; // остальные символы запрещены
     end;
end;

// EditChange - текст, находящийся в поле редактирования,
// изменился. Процедура EditChange обрабатывает изменение
// текста в полях Курс и Цена
```

```
procedure TForm1.EditChange(Sender: TObject);
begin
    // проверим, есть ли данные в полях редактирования
    if (Length(Edit1.Text) = 0) or (Length(Edit2.Text) = 0)
       // кнопка Пересчет недоступна
       then Button1.Enabled := False
       // кнопка Пересчет доступна
       else Button1.Enabled := True;
end;

// щелчок на кнопке Пересчет
procedure TForm1.Button1Click(Sender: TObject);
var
  usd: real;   // цена в долларах
  k:   real;   // курс
  rub: real;   // цена в рублях

begin

    k := StrToFloat(Edit1.Text);
    usd := StrToFloat(Edit2.Text);

    // пересчитать цену из долларов в рубли
    rub := usd * k;

    // вывести результат расчета в поле Label4
    Label4.Caption := FloatToStr(usd) + '$ = ' +
                      FloatToStrF(rub, ffCurrency, 6,2);
end;
```

5. Написать программу, которая пересчитывает вес из фунтов в килограммы (1 фунт = 409,5 грамм). Рекомендуемый вид формы приведен на рис. 1.3. Программа должна быть спроектирована таким образом, чтобы пользователь мог ввести в поле **Вес в фунтах** только положительное число (целое или дробное).

6. Написать программу, которая вычисляет скорость (км/час), с которой бегун пробежал дистанцию. Рекомендуемый вид формы приведен на рис. 1.4. Программа должна быть спроектирована таким образом, чтобы в поля **Дистанция** и **Минут** можно было ввести только целое число, а в поле **Секунд** — дробное.

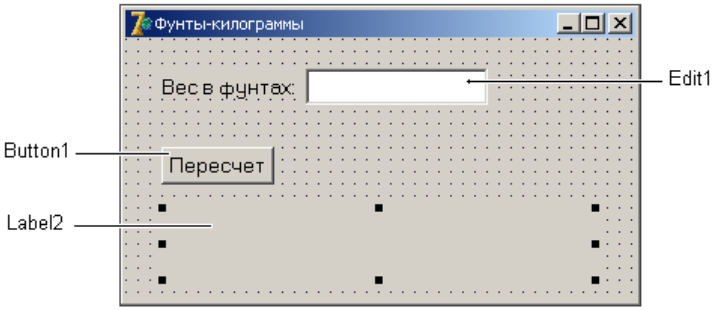

Рис. 1.3. Форма программы **Фунты-килограммы**

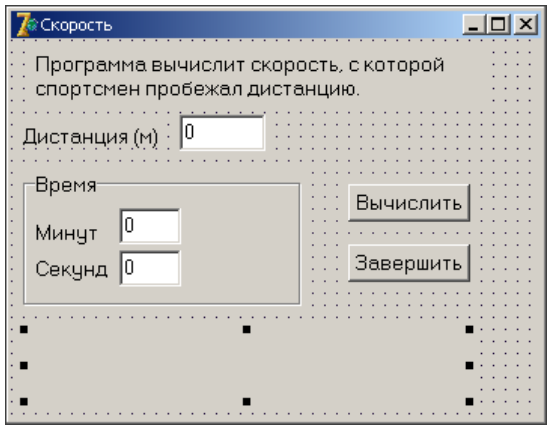

Рис. 1.4. Форма программы **Скорость**

```
// нажатие клавиши в поле Дистанция
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
    // Key - символ, соответствующий нажатой клавише.
    // Если символ недопустимый, то процедура заменяет его
    // на символ с кодом 0. В результате этого символ в поле
    // редактирования не появляется и у пользователя создается
    // впечатление, что программа не реагирует на нажатие
    // некоторых клавиш.
    case Key of
      '0'..'9':              ; // цифра
      #8    :                ; // <Backspace>
      #13   : Edit2.SetFocus; // <Enter> - курсор в поле Минут
```

```
      // остальные символы запрещены
      else Key :=Chr(0);  // символ не отображать
    end;
end;

// нажатие клавиши в поле Минут
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
    case Key of
      '0'..'9':            ;
      #8   :               ; // <Backspace>
      #13  : Edit3.SetFocus; // <Enter> - курсор в поле Секунд
      // остальные символы - запрещены
      else Key :=Chr(0);  // символ не отображать
    end;
end;

// нажатие клавиши в поле Секунд
procedure TForm1.Edit3KeyPress(Sender: TObject; var Key: Char);
begin
   case Key of
      '0'..'9': ;
      ',','.' : // десятичный разделитель
               begin
                  Key := DecimalSeparator;
                  if Pos(DecimalSeparator,Edit3.Text) <> 0
                     then Key := Char(0);
                end;
      #8:  ; // <Backspace>
      #13: Button1.SetFocus; // фокус на кнопку Вычислить
      // остальные символы - запрещены
      else Key :=Chr(0);  // символ не отображать
    end;
end;

// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
var
    dist : integer; // дистанция, метров
    min : integer;  // время, минуты
    sek : real;     // время, секунды
    v: real;        // скорость
```

```
begin
    // Если поле редактирования не содержит данных,
    // то при выполнении преобразования строки в число
    // (функция StrToInt или StrToFloat)
    // возникает исключение EconvertError.
    // Чтобы предотвратить эту ситуацию,
    // проверим, есть ли данные в полях редактирования
    // и, если их там нет, запишем нулевое значение.

    if Length(Edit1.Text) = 0
        then Edit1.Text := '0';

    if Length(Edit2.Text) = 0
        then Edit2.Text := '0';

    if Length(Edit3.Text) = 0
        then Edit3.Text := '0';

    // получить исходные данные из полей ввода
    dist := StrToInt(Edit1.Text);
    min  := StrToInt(Edit2.Text);
    sek  := StrToFloat(Edit3.Text);

    // дистанция и время не должны быть равны нулю
    if (dist = 0) or ((min = 0) and (sek = 0)) then
    begin
        MessageDlg('Надо задать дистанцию и время',
                   mtWarning,[mbOk],0);
        exit;
    end;

    // вычисление
    v := (dist/1000) / ((min*60 + sek)/3600);

    // вывод результата
    label5.Caption := 'Дистанция: '+ Edit1.Text + ' м' + #13 +
                      'Время: ' + IntToStr(min) + ' мин ' +
                       FloatToStrF(sek, ffFixed,4,2) + ' сек' +
                       #13 + 'Скорость: ' +
                       FloatToStrF(v,ffFixed,4,2) + ' км/час';
end;
```

```
// щелчок на кнопке Завершить
procedure TForm1.Button2Click(Sender: TObject);
begin
        // закрыть главную форму - завершить работу программы
        Form1.Close;
end;
```

7. Написать программу, которая вычисляет силу тока в электрической цепи. Рекомендуемый вид формы приведен на рис. 1.5. Программа должна быть спроектирована таким образом, чтобы кнопка **Вычислить** была доступна только в том случае, если пользователь ввел величину напряжения и сопротивления.

Рис. 1.5. Форма программы **Сила тока**

8. Написать программу, которая вычисляет сопротивление электрической цепи, состоящей из двух параллельно соединенных резисторов. Рекомендуемый вид формы приведен на рис. 1.6.

9. Написать программу, которая вычисляет доход по вкладу методом простых процентов (Доход = Сумма * Процент / 12 * Срок). Рекомендуемый вид формы программы приведен на рис. 1.7. В результате щелчка на кнопке **Вычислить** в окне программы должна отображаться величина дохода и сумма в конце срока вклада. Программа должна быть спроектирована таким образом, чтобы в поля **Сумма** и **Процентная ставка** можно было ввести дробные числа, а в поле **Срок** — только целое.

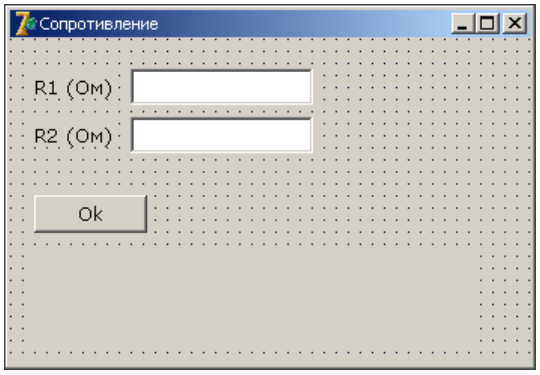

Рис. 1.6. Форма программы **Сопротивление**

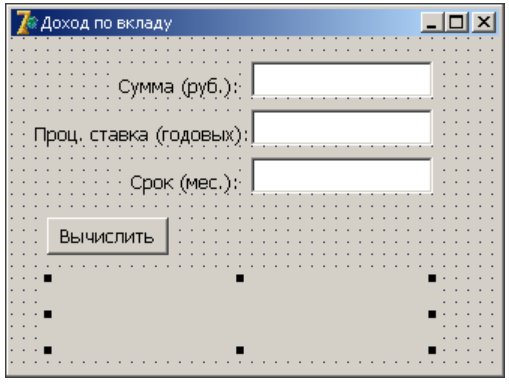

Рис. 1.7. Форма программы **Доход по вкладу**

10. Написать программу, которая вычисляет сопротивление электрической цепи, состоящей из двух сопротивлений. Сопротивления могут быть соединены последовательно или параллельно. Рекомендуемый вид формы приведен на рис. 1.8. Если величина сопротивления цепи превышает 1 000 Ом, то результат должен быть выведен в килоомах.

```
// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
```

```
var
    r1,r2: real; // величины сопротивлений
    r: real;     // сопротивление цепи
begin
    // получить исходные данные
    r1 := StrToFloat(Edit1.Text);
    r2 := StrToFloat(Edit2.Text);

    if (r1 = 0) and (r2 = 0) then
    begin
        ShowMessage('Надо задать величину хотя бы одного
                     сопротивления');
        exit;
    end;

    // переключатели RadioButton1 и RadioButton2
    // зависимые, поэтому о типе соединения можно
    // судить по состоянию одного из них
    if RadioButton1.Checked
        then // выбран переключатель Последовательно
            r:= r1+r2
        else //  выбран переключатель Параллельно
            r:= (r1*r2)/(r1+r2);

    Label4.Caption := 'Сопротивление цепи: ';
    if r < 1000 then
       Label4.Caption := Label4.Caption +
                  FloatToStrF(r,ffFixed,3,2) + ' Ом'
    else
        begin
            r:=r/1000;
            Label4.Caption := Form1.Label4.Caption +
                  FloatToStrF(r,ffFixed,3,2) + ' кОм';
        end
end;

// щелчок на переключателе Последовательно
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
   // пользователь изменил тип соединения
   Label4.Caption := '';
end;
```

```
// щелчок на переключателе Параллельно
procedure TForm1.RadioButton2Click(Sender: TObject);
begin
   // пользователь изменил тип соединения
   Label4.Caption := '';
end;
```

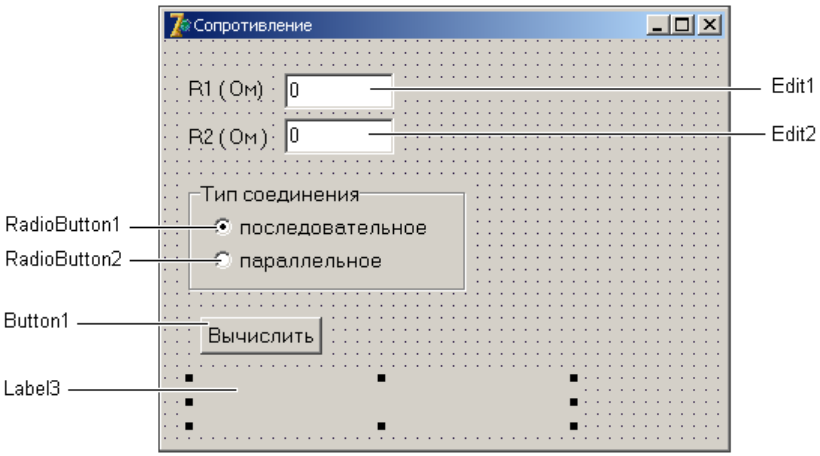

Рис. 1.8. Форма программы **Сопротивление**

11. Напишите программу, которая вычисляет доход по вкладу. Программа должна обеспечивать расчет простых и сложных процентов. Простые проценты начисляются в конце срока вклада, сложные — ежемесячно и прибавляются к текущей (накопленной) сумме вклада, в следующем месяце проценты начисляются на новую сумму. Рекомендуемый вид формы программы приведен на рис. 1.9.

```
// щелчок на кнопке Вычислить
procedure TForm1.Button1Click(Sender: TObject);
var
    sum : real;       // сумма вклада
    pr:   real;       // процентная ставка
    period: integer; // срок вклада
    profit: real;     // доход по вкладу
```

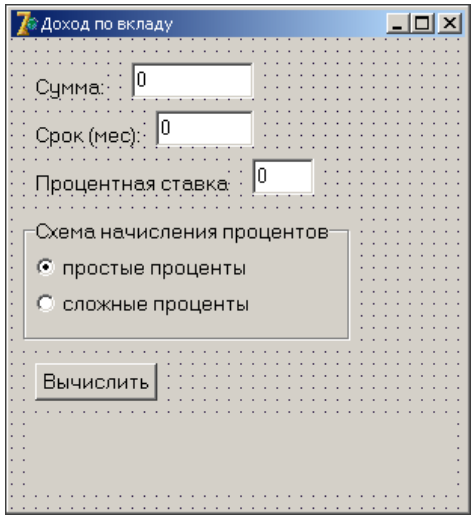

Рис. 1.9. Форма программы **Доход по вкладу**

```
    // сумма при вычислении методом сложных процентов
    sum2: real;
    i: integer;
begin
    // получить исходные данные
    sum := StrToFloat(Edit1.Text);
    pr := StrToFloat(Edit2.Text);
    period := StrToInt(Edit3.Text);

    if RadioButton1.Checked then
        // выбран переключатель Простые проценты
        profit := sum * (pr/100/12) * period

    else
        // т.к. в группе два переключателя, если
        // не выбран RadioButton1, то выбран
        // RadioButton2 - Сложные проценты
        begin
            sum2:= sum;
            for i:=1 to period do
                sum2 := sum2 + sum2 * (pr/100/12);
```

```
            // здесь sum2 - сумма в конце срока вклада
            profit := sum2 - sum;
        end;

    sum := sum + profit;
    Label4.Caption := 'Доход: ' +
                      FloatToStrF(profit,ffCurrency,6,2) + #13 +
                      Сумма в конце срока вклада: ' +
                      FloatToStrF(sum,ffCurrency,6,2);
end;
```

12. Написать программу, которая вычисляет стоимость жалюзи. Рекомендуемый вид формы приведен на рис. 1.10.

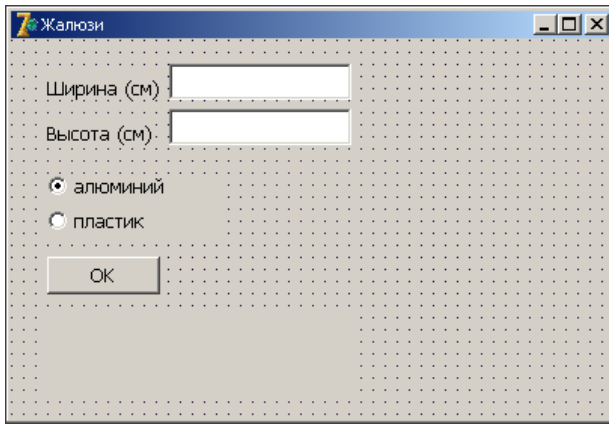

Рис. 1.10. Форма программы **Жалюзи**

13. Написать программу, которая позволяет пересчитать цену из долларов в рубли или из рублей в доллары. Рекомендуемый вид формы приведен на рис. 1.11. Во время работы программы, в результате выбора вида конвертации, соответствующим образом должен меняться заголовок окна и текст, поясняющий назначение полей ввода.

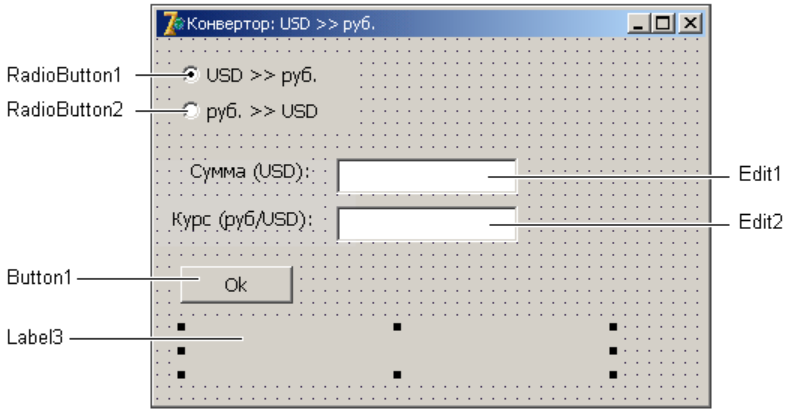

Рис. 1.11. Форма программы **Конвертор**

```
  // щелчок на переключателе  USD >> руб.
procedure TForm1.RadioButton1Click(Sender: TObject);
begin
  // изменить заголовок окна
  Form1.Caption := 'Конвертор: USD >> руб.';

  // изменить текст перед полем Edit1
  Label1.Caption := 'Сумма ($): ';

  // установить курсор в поле Цена
  Edit1.SetFocus;

  Label3.Caption := '';
end;

// щелчок на переключателе  руб. >> USD
procedure TForm1.RadioButton2Click(Sender: TObject);
begin
  Form1.Caption := 'Конвертор: руб. >> USD';
  Label1.Caption := 'Сумма (руб.): ';
  Edit1.SetFocus;
  Label3.Caption := '';
end;
```

```
// Щелчок на кнопке OK
procedure TForm1.Button1Click(Sender: TObject);
var
  usd: real; // цена в долларах
  rub: real; // цена в рублях
  k: real;   // курс

begin

  k := StrToFloat(Edit2.Text);

  if RadioButton1.Checked then
    begin
      // пересчет из долларов в рубли
      usd := StrToFloat(Edit1.Text);
      rub := usd * k;
      Label3.Caption := FloatToStrF(usd,ffFixed,6,2) +
              '$ = ' + FloatToStrF(rub,ffCurrency,6,2);
    end
  else begin
      // пересчет из рублей в доллары
      rub := StrToFloat(Edit1.Text);
      usd := rub / k;
      Label3.Caption := FloatToStrF(rub,ffCurrency,6,2) +
              ' = ' + FloatToStrF(usd,ffFixed,6,2) + '$';
  end;
end;

// процедура обрабатывает событие EditChange
//  компонентов Edit1 и Edit2
procedure TForm1.EditChange(Sender: TObject);
begin
  // если в каком-либо из полей Edit нет данных,
  // сделать кнопку Button1 недоступной
  if (Length (Edit1.Text) = 0) or  (Length (Edit2.Text) = 0)
     then Button1.Enabled := False
     else Button1.Enabled := True;
  Label3.Caption := '';
end;
```

```
// нажатие клавиши в поле Сумма
procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
  case Key of
       '0'..'9', #8: ;  // цифры и <Backspace>
       '.',',':
            // Обработку десятичного разделителя
            // сделаем "интеллектуальной". Заменим точку и
            // запятую на символ DecimalSeparator - символ,
            // который при текущей настройке операционной
            // системы должен использоваться
            // при записи дробных чисел.
            begin
               Key := DecimalSeparator;
               // проверим, введен ли уже в поле Edit
               // десятичный разделитель
               if pos(DecimalSeparator,Edit1.Text) <> 0
                  then Key := #0;
            end;
        #13: // клавиша <Enter>
             // Переместить курсор в поле Курс
             Edit2.SetFocus
        else Key := #0; // остальные символы запрещены
    end;
end;

// нажатие клавиши в коле Курс
procedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
begin
  case Key of
       '0'..'9', #8: ;  // цифры и <Backspace>
       '.',',':
           begin
               Key := DecimalSeparator;
               if pos(DecimalSeparator,Edit1.Text) <> 0
                  then Key := #0;
            end;
        #13: // клавиша <Enter>
             // Переместить фокус на кнопку Ok
             Edit2.SetFocus
```

```
        else Key := #0; // остальные символы запрещены
    end;
end;
```

14. Написать программу, которая вычисляет доход по вкладу сроком на 1, 2, 3, 6 месяцев или на один год (предполагается, что процентная ставка зависит от срока вклада). Рекомендуемый вид формы приведен на рис. 1.12.

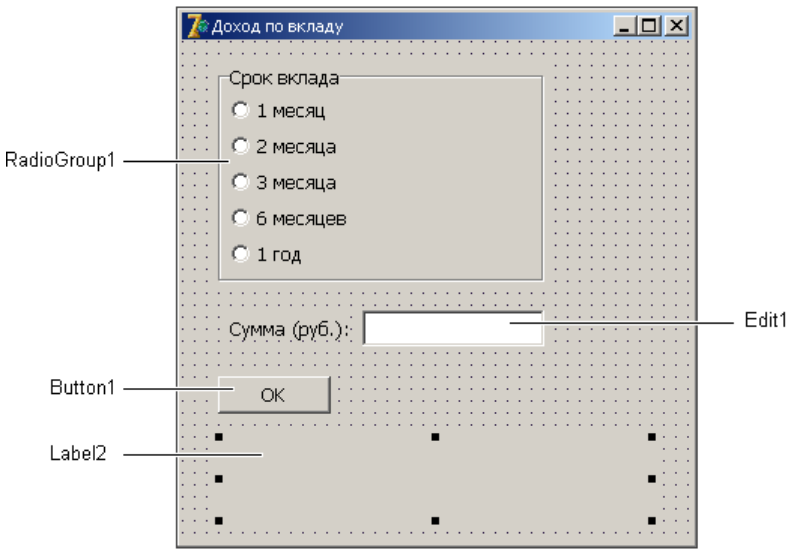

Рис. 1.12. Форма программы **Доход по вкладу**

```
// щелчок на кнопке OK
procedure TForm1.Button1Click(Sender: TObject);
var
  sum: real;         // сумма вклада
  period: integer;   // срок вклада (месяцев)
  percent: real;     // процент (годовых)
  profit: real;      // доход
  sum2: real;        // сумма в конце срока вклада

begin
  sum := StrToFloat(Edit1.Text);
```

```
  case RadioGroup1.ItemIndex of
    0: begin
        period := 1;
        percent := 8;
       end;
    1: begin
        period := 2;
        percent := 8.5;
       end;
    2: begin
        period := 3;
        percent := 9;
       end;
    3: begin
        period := 6;
        percent := 10 ;
       end;
    4: begin
        period := 12;
        percent := 11;
       end;
   end;

   profit := sum * percent/100/12 * period;
   sum2 := sum + profit;

   Label2.Caption := 'Сумма вклада: ' +
              FloatToStrF(sum, ffCurrency,6,2) + #13 +
             'Срок вклада: ' + IntToStr(period) + 'мес.' +
              #13 + 'Процентная ставка: ' +
              FloatToStrF(percent, ffFixed,6,2) + '%' + #13 +
             'Доход: ' + FloatToStrF(profit, ffCurrency,6,2);
end;

procedure TForm1.Edit1KeyPress(Sender: TObject; var Key: Char);
begin
  case Key of
       '0'..'9', #8: ;  // цифры и <Backspace>

       '.',',':
            // Обработку десятичного разделителя
            // сделаем "интеллектуальной". Заменим точку и
```